Different approaches for text entry only via gaze

THOMAS BECKER, RWTH Aachen
JANNIK HELLENKAMP, RWTH Aachen

We will introduce the reader to the topic of gaze interaction. Furthermore, we will give challenge orientated an overview of five approaches for gaze interaction. These approaches are Dasher, 9-direction Gaze Estimation, EyeSwipe, pEyes and iType. In the end we will discuss and compare them.

ACM Reference Format:
Thomas Becker and Jannik Hellenkamp. 2020. Different approaches for text entry only via gaze. 1, 1 (January 2020), 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Gaze interaction describes the interaction with a device only using the user's gaze. People with physical restrictions caused by a disability for instance can use gaze interaction as an input method for a device. Gaze interaction offers new ways to interact with devices. It can support disabled people in interacting with their environment. For example people that are not able to speak and to use their hands, can use gaze to interact with a device to do eye typing [3].
It is also interesting because of privacy reasons. Sensible personal data can better be protected, for instance typing the password into the smartphone in a public space. Nearby people could watch on your screen and recognize it. With gaze typing that is impossible [4].

1.1 Challenges

With all of these new approaches come a lot of possibilities for users but also a lot of challenges [2]. We will introduce in some common challenges.

1.1.1 Dwell time. One of the most common problems with gaze interaction is, that the user misses an action emitter. This could be a button, a blink with the eye or any other none gaze interaction. The term *Dwell time* describes a solution for this problem. The user emits an event by keeping his gaze on the same spot for a defined period of time. This duration is called *dwell* time. This technique can slow down the speed of the typing [3].

1.1.2 Noisy Data. There are a lot of reasons why data can get noisy[7]. The main three reasons for noisy data are:

(1) The human gaze view point is not focused on one concrete point. A human will always have a little angle that he is looking at.
(2) Moving of the head can also lead to noisy data, because this little movement also influences the eye positions.

Authors' addresses: Thomas Becker, RWTH Aachen, thomas.becker1@rwth-aachen.de; Jannik Hellenkamp, RWTH Aachen, jannik.hellenkamp@rwth-aachen.de.

XXXX-XXXX/2020/1-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

(3) The human eye is doing saccade and involuntary blinking. Saccades are very short fast movements of the gaze view point, which are subconscious. But this jiggling is tracked by eye trackers and this leads to noisy data.

1.1.3 Lighting Conditions. Bad lighting conditions can lead to wrong data and issues with the processing of the wrong data. This can lead to wrong decisions of the algorithm. The data images can be blurred or the eye tracker is not able to detect where the eyes are looking at.

1.1.4 Complexity for novel users. A good gaze interaction system should be simple and clear to understand. Especially for novel users a complex system could make the interaction not easier. It will make it worse. So it is important to keep care of the complexity of the interaction method.

1.1.5 Different Interfaces and Input methods. Another common problem is that there already are a lot of gaze interaction systems and each system has a different input method and also a different interface. Users that want to interact with the device via gaze have different steps as well as different interfaces for each application (e.g. desktop navigation and gaze typing). This can lead to confusion, if the user needs to remember the interaction sequence for each different application they want to use [1].

2 DIFFERENT SYSTEMS

2.1 Dasher

Dasher is an eye-typing system. It was originally presented by Ward and MacKay in 2002. The system is a dwell time free approach. [6]

2.1.1 How to type with Dasher. While Dasher is most known to work with an eye tracker, the system can be used with a variety of input devices. The input for Dasher is a two-dimensional pointing device. This could be a mouse, a joystick or a set of buttons. In order for Dasher to be used with gaze, the input is an eye tracking device.

The user starts the input with an ordered alphabet on the right side of the screen as shown in Figure 1. Each character is surrounded by an equally sized box. To start with the typing, the user moves the cursor in the direction of the character, that he wants to select. When the cursor is right to the center line, the system will start to zoom into the pointed direction. Also the box around the pointed character will start to grow. A character is selected, when it passes the center line. When the user selects a character, the system predicts the probability of being selected for each character. The system then scales the boxes around the characters corresponding to their probability as shown in Figure 2. The written text appears in a text box above the canvas.

2.1.2 How to correct the input. If the user accidentally selects the wrong character, he can shift the controller to the left side of the center line. This will zoom out of the past character and the character moves back to the right side of the canvas. Now the user can select another character by moving the cursor back to the right side of the center line towards the correct character.

2.1.3 User study. The authors conducted a user study with 12 participants. In the first session with dasher, the users had a average for text entry of 2.49 words per minute (wpm). This increased to 17.26 wpm in the tenth session. The corrections to typed characters ratio was 0.26 in the first session and 0.13 in the tenth session.

2.1.4 Problems. While Dasher solves the dwell time challenge, the system does not handle the actual eye tracking process. This includes the challenges of noisy data and bad lighting conditions. Furthermore, it is hard to use for novel users, caused by the speed of dasher.

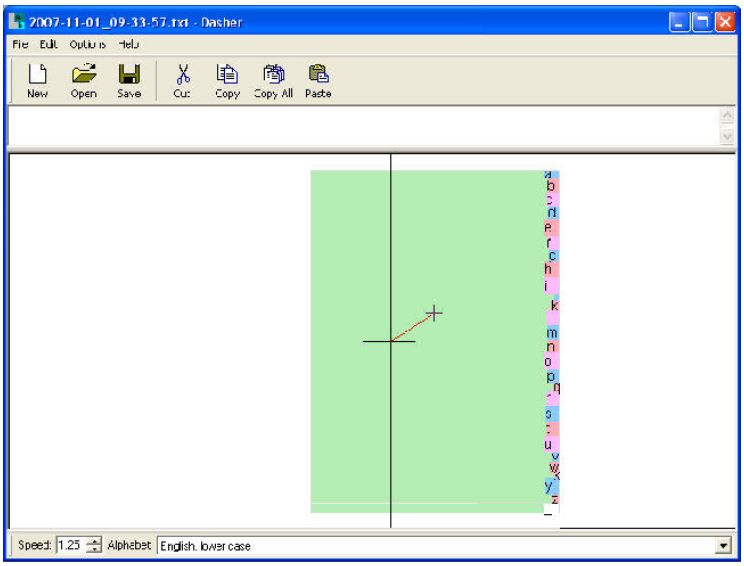

Fig. 1. The start screen of Dasher [6]. You can see the gaze target represented with the cross in the middle. Also there is the center line and on the right side you can see the letters alphabetically ordered from top to bottom surrounded by a border

Additionally, if the user wants to see what he typed, he needs to look to the left side of the screen, this can delete already typed letters.

2.2 9-direction Gaze Estimation

Gaze typing also needs the eye tracking process as an input. The authors try to also solve the challenges of the eye tracking part [7]. These challenges are noisy data, by saccade, the biological restrictions of an eye and its focus and involuntary blinking, and also the long calibration time and different lighting conditions.
This is an approach which divides the gaze of a human into nine directions. The user can select a target via voluntary blinking and this technique uses a T9 Keyboard surface to order the letters.
This method uses a convolutional neural network (CNN) in combination with the nine direction screen to solve these problems.
Furthermore, there are two modes to use - the screen mode and the off-screen mode. During the screen mode there is a screen and also a camera attached to the screen. During the off screen mode there is no screen and the users get the camera directly on their head. On the one hand this can lead to problems with novel users because they need to remember the positions where each letter is. On the other hand people are free to use it everywhere.

2.2.1 The calibration process. Normal eye trackers need a long time to calibrate. Targets are shown on the display and the user needs to focus them with gaze. Head movement or also subconscious body movement can distort the calibration process. A lot users from gaze interaction are disabled and so that is a problem for them.
With the 9-direction technique just 10 to 20 seconds are needed to calibrate. This is possible because of the CNN. The authors trained the CNN with thousands of pictures to estimate the gaze of the user.

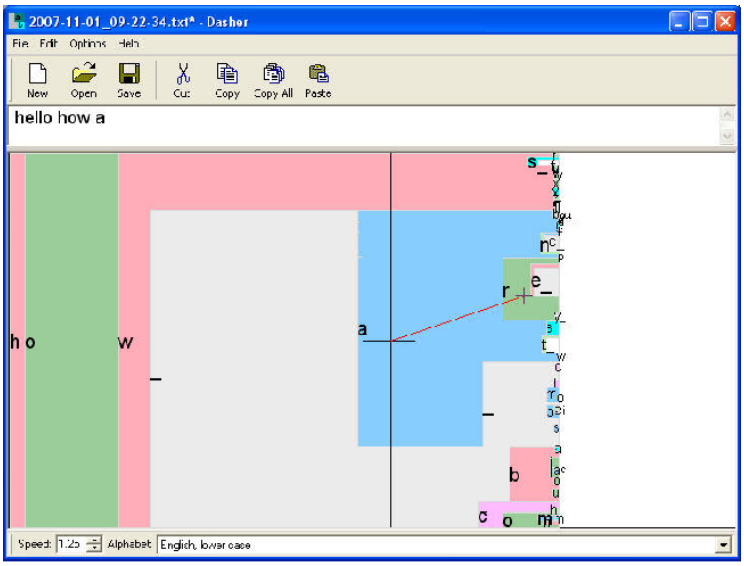

Fig. 2. The Dasher software in action [6]. The user types "hello how are you", using his gaze target shown by the cross.

They also used different light levels. So the user does not need to be in a laboratory for calibration and usage.

2.2.2 The Interface. The whole system has ten states: left-up, middle-up, right-up, left, middle, right, left-down, middle-down, right-down and closed eyes. The screen is ordered like the T9-keyboard from old bar phones (see Figure 3 (a)). That is a huge advantage for gaze interaction, because the targets are bigger compared to a QWERT keyboard. That is important because the human eye never focuses one exact point. The viewpoint has an angle of approximately 0.5 degrees. The error rate is extremely reduced because of this screen style. Also it fits with the nine directions of the gaze.

2.2.3 Handling noisy data. The human eye makes saccades and blinks involuntarily, which distorts the results of a common eye tracker. This technique implemented a robustness function to prevent this. 16 frames are stored in an array and to select a target there needs to be at least eight following frames focusing this one target. A saccade lasts less than four frames in general and also an involuntary blinking lasts less than five frames.
9-direction gaze estimation also uses both eyes to estimate the gaze. Using just one eye can lead to different problems. First, one eye is more sensitive to head movements. Second, it can always happen that the resolution is too bad to estimate the gaze, caused by blur or bad lighting conditions. If you use two eyes the algorithm can choose the other one. Third, if the target is too close to the eyes, the eyes will not be synchronized in their gaze direction.

2.2.4 The typing process. To type the user needs to select the right direction with his eyes and gaze and blink voluntarily. Then the user gets a feedback from the system (sound in off-screen mode and sound plus visual feedback in screen mode) and the system switches to another state, where now the letters are displayed, also ordered in the nine directions. This is shown in Figure 3.

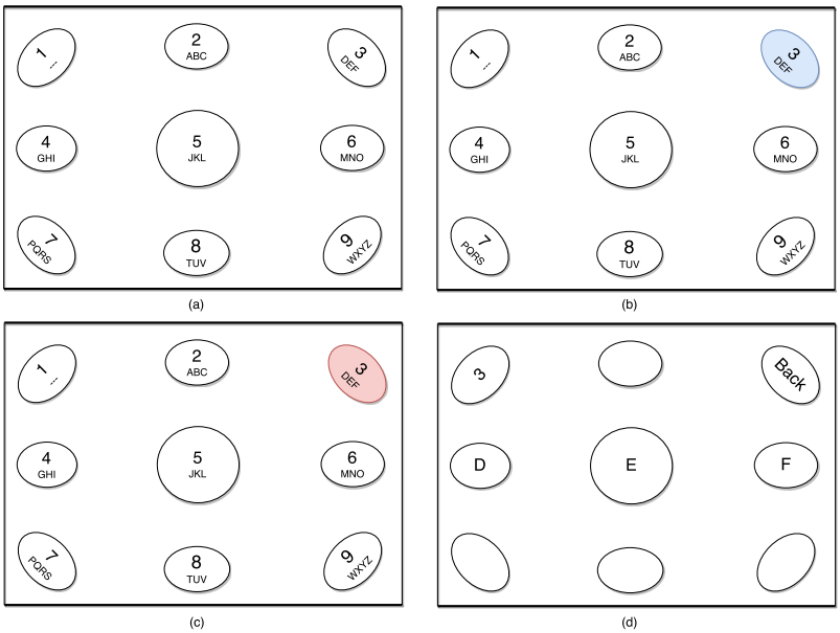

Fig. 3. The eye typing process of 9-direction gaze estimation [7]. The normal interface (a). The user selects a target (b). Selection via blinking (c). Stepping into next interface with the letters (d).

2.2.5 User study. The authors also conducted a user study with beginners and experts. The main results are that they could reduce the error rate significantly to less than 0.05 errors in mean. The users did up top 20 wpm in screen mode and up to 18 wpm in off-screen mode. If a language model is combined with their method then theoretically the maximum is about 40 wpm.

2.2.6 Problems. This approach solves the challenge of noisy data and bad lighting conditions, but the system can be confusing for novel users. A new user needs to know the position of the characters. This is especially a problem in the off-screen mode. In the following section we present a more intuitive technique.

2.3 EyeSwipe

EyeSwipe is a dwell-free eye typing method from 2016 [3]. It uses the gaze path of the user as input. It is based on the already existing method of swiping one's finger through a touch screen keyboard. EyeSwipe just replaces the use of the finger with the use of the gaze. This causes some problems to come up. One is about how the user selects the start and the end of a word. To solve that EyeSwipe uses the method of „reverse crossing", we will explain that in the following section. The purpose of this technology is to give users an easy and comfortable and also fast way to interact with the device.

2.3.1 How to operate with EyeSwipe. We already mentioned that EyeSwipe is based on the idea of the method of swiping finger through a touch screen keyboard just replacing the finger with gaze and using the gaze path as input.
To start the typing process on a touch screen keyboard you just need tap on the screen, hold it and swipe. Here the gaze view point is normally already on the display. That is why EyeSwipe is using the *reverse crossing* method to select the start and end letter of a word. So initially the user needs to

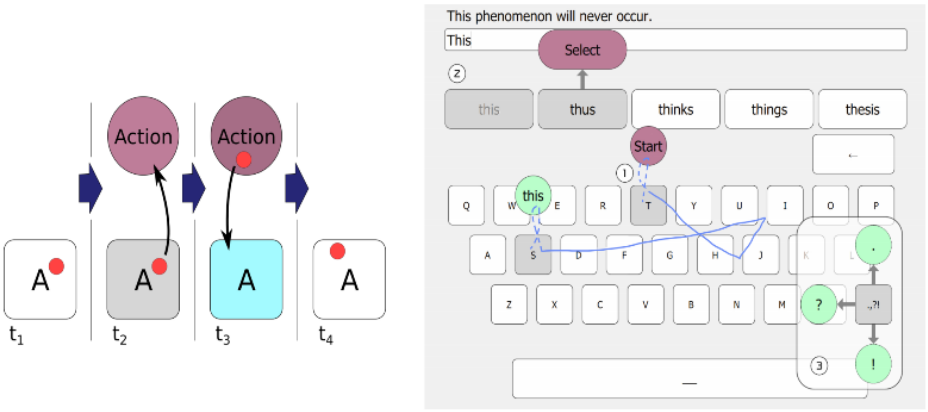

Fig. 4. There is the reverse crossing sequence (left) and the swiping of the gaze path with the purposed candidates (right) illustrated. Both pictures are taken from the EyeSwipe paper [3].

look directly on the target letter. Then a button, titled with e.g. „start gaze path“, will appear above the letter. The user needs to look at least 100 ms on this button and then back on the letter to start the gaze path. Now the user can simply swipe with the gaze over the screen and it is detected as input (see Figure 4). During the gaze path swiping the user does not even need to look directly on the target key, it is enough to look in the vicinity of the key. EyeSwipe is calculating possible meaningful word candidates and purposes based on this information these are shown to the user later on (see Figure 4).
To select the last letter of a word the user needs to hold the gaze viewpoint on the target; a button will appear again, titled with the best candidate, calculated by EyeSwipe based on the gaze path. If the user then looks at least 100 ms at this button, it is detected as this word.

2.3.2 Candidates Selection. That is an important part of this input method, because EyeSwipe's purpose is to be fast. So the users mostly will not look at the center of each key. The user should swipe through the keyboard with the gaze. So EyeSwipe needs to calculate possible candidates based on the noisy data of the gaze path of the user.
The calculation of these candidates is based on the principle of the ideal path. The ideal path of a word is describing the center points of the each letter of the word. So for example the ideal path of the word „eva“would be the center of the key „e“, „v“and „a“.
EyeSwipe is now comparing the input gaze path with the ideal paths saved in the data base using Dynamic Time Warping. This is a common technique to compare to sequences. This is also used in the touch swiping applications.
Eyeswipe also uses an average filter to filter the noisy data, because the ideal path has on average less than ten key points but the gaze input is quite noisy with a lot points close to other letters that do not belong to the word, that the user wants to type in.
In the end EyeSwipe suggests the best candidates to the user shown on the screen and the user can select what he wanted to type. If a word is not already part of the data base, the user can add it to the data base.

2.3.3 User study. The authors conducted EyeSwipe in a user study. To compare the results they used a dwell-time based one as a second approach. They did four sessions with ten participants for EyeSwipe and also four for the dwell-time approach. Each session was ten minutes long with a

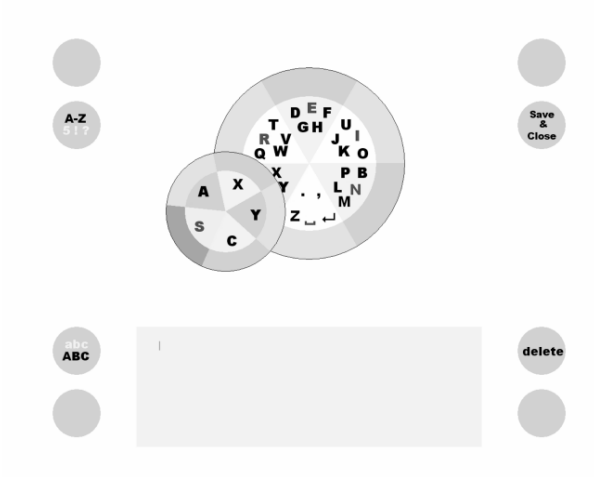

Fig. 5. The pEYEs interface [1]. We can see the main pie in the background and also an already selected part, shown here via the second pie in the foreground. There are also some settings buttons on the sides. The text field is at the bottom.

short break after it. With EyeSwipe the participants reached an average typing rate of 11.7 wpm and only 9.5 wpm with the dwell-time based approach. The participants said that EyeSwipe was more efficient, especially for longer words.

2.3.4 Problems. EyeSwipe is not completely dwell time free. The activation and deactivation of the typing process of one word needs dwell time. This slows the input speed down. Moreover, the technology can only be used for text production applications and is not universal.

2.4 pEYEs

In contrast to the technologies presented before, pEYEs is an eye typing technology focused on unifying interfaces for gaze control [1].
It was presented in 2007, where already a lot of approaches for gaze control are presented, but all have different interfaces and also are different in the form of using them. That is especially confusing for novel users and so it slows down the interaction. Assume a disabled person, using gaze control to interact with devices, but this person has ten different applications running on this device and each one has different interfaces and different steps to do for interacting. pEYEs tries to simplify this.
So pEYEs is mainly focused on how the interface of an application for gaze control looks and which steps need to be done.

2.4.1 The Interface. The Main idea of pEYEs is to order the targets for the gaze as a pie. This pie is divided into six sections filled with the targets, for example the targets for the gaze typing shown in Figure 5. Here the targets are letters. The placing of the letters is important here. The authors marked the five most frequent letters in red and also they arranged the space symbol in a way that users can easily access this target in one saccade. That speeds up the typing process.

2.4.2 How to select Targets. If the user wants to select a target, he needs to select via his gaze and confirm with a tap to the outer ring of the pie section in which the target is.

After that a second pie pops up, in which the targets of the selected section are presented, again in a pie shape, but this time divided in five sections as shown in Figure 5. The keyboard requires a dwell time (set to 400 ms) to select a target.

2.4.3 User Study and Results. The authors evaluated the system in a user study in which they compared the pEYEs approach with a simple ABCDE keyboard. For that they used beginners and also expert users.
They investigated mainly two areas.

(1) **Speed** This is measured in words per minute. Here novel users had higher wpm rates with the ABCDE keyboard then with pEYEs as you can see in Figure 6. But the experts got nearly the same rate with pEYES and the ABCDE keyboard.

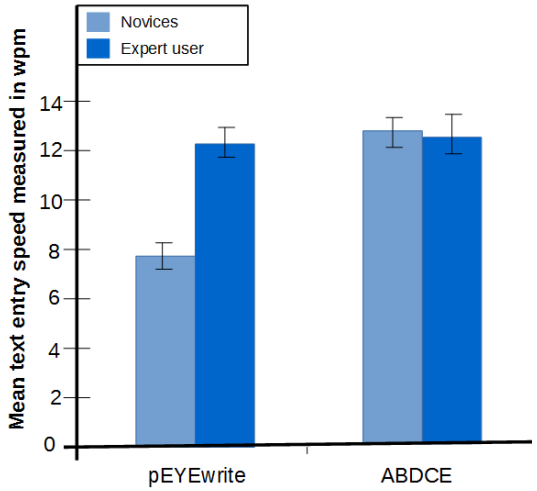

Fig. 6. The speed results of the user study. You can see, that novice users had higher wpm mean rates with the ABCDE keyboard and experts had nearly the same wpm rate with both approaches. This diagram is based on the diagram from the authors [1].

(2) **Accuracy** This is measured in keystrokes per character (KSPC). This increases by typing errors. If the user is doing a lot of mistakes by choosing the wrong letter, the KSPC will increase.
Novice users started in general with 1.184 KSPC and could reduce this value during the study to 1.06 KSPC. So they reduced their input overhead from 11% to 6%. In comparison to that, the novice users decreased their KPSC with the ABDCE keyboard from 1.16 to 1.08, so about 8%.
The expert users could reduce their KPSC with pEYEs from 1.109 to 1.0 (typo free). With the ABDCE keyboard the KSPC rate did not change.

Summing up, we can see that the speed of users, that are familiar with pEYEs is around 12 wpm and pEYEs is offering lot potential in accuracy. Users can typo free type via pEYEs if they train.

2.5 iType

While most of the presented systems where developed to work with complete sentences, the iType system has a different background. It was developed to input short sequences of sensitive

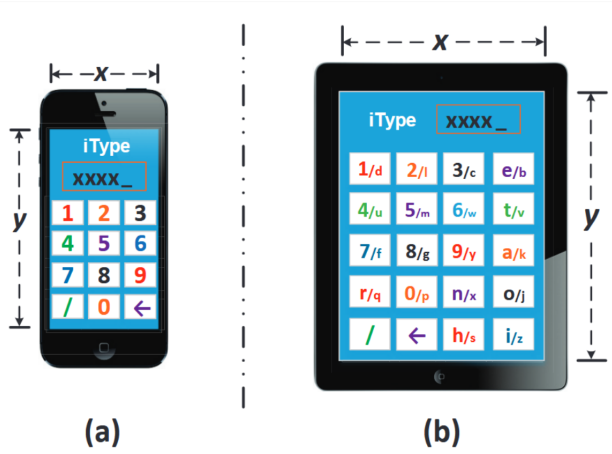

Fig. 7. The button layout for iType[4]. (a) shows the layout for numerical input and (b) shows the layout for alphanumerical input

information (passwords, credit card information) on a mobile device [4]. The advantage of such an input method is, that the user can input a password in the public and other people can not watch his fingers and steal his information.

2.5.1 Typing with iType. iType works with the selfie camera built into most modern mobile phones. The user looks at a screen, where he can see a grid of buttons. These buttons can either be numbers or the alphabet split up into two pages as shown in Figure 7. To select a number or a character, the user has to shift his gaze onto the selected button. While the iType system does not specify a dwell time, the button will be selected when the gaze does not shift for some time. One of the key difference to the other technologies is, that the system can not predict the next character, because passwords tend to be random.

2.5.2 User study. The iType system has been tested with passwords that where 5-8 characters long. The authors of the user study argue that the password does not need to be long, when it is typed via gaze. The accuracy of the system is 97% in a static environment and 89% in a dynamic environment. The time between each keystroke is 2.0*s* in static and 2.6*s* in a dynamic environment. The overall experience was also measured. In 84% of the cases, the password was typed in correctly on the first attempt and all passwords were typed in correctly after two attempts.

2.5.3 Problems. The biggest problem with iType is that the user can not see, what he is typing. While this is necessary for the privacy and safety of the password, it is not so user friendly.

3 COMPARISON

To compare the five different techniques, we are comparing the results of the user studies. This comparison should be treated with some caution, because some techniques, especially iType, were developed to solve a problem for another use case for eye typing. With that in mind, the first comparative aspect will be the words per minute (wpm).

3.1 Words per minute

When following the results of the user studies the fastest method is 9-direction Gaze Estimation. The user study evaluates a mean words per minute (wpm) count of 19 in the third session. The results of the study is based on only 5 participants. Also there was a training in between the sessions. Dasher also has a fast wpm count of 17.26 after 10 sessions. 12 participants took part in this study and there was no training in-between sessions. EyeSwipe and pEYEs both have a lower wpm of 11.7 (EyeSwipe) and 12 (pEYEs). There is no wpm value for iType, because the software is not designed to input words.
To put those values in perspective, the average wpm on a smartphone is 36.2 wpm. [5]

3.2 Keystrokes per character (KSPC)

Another comparative aspect is the amount of keystrokes per character (KSPC). This describes the amount of keys the user needs to press to get the correct character. If the user makes a lot of mistakes, the KSPC increases. With pEYEs a novice user started with an KSPC of 1.184 and they improved to a KSPC of 1.06.
With Dasher, the KSPC is useless, because there are no keystrokes. Here, we look at the ratio of typed characters to corrected characters. This ratio is 0.26 corrections per typed character in the first session and 0.13 in the tenth session.

4 SUMMARY

We have seen five approaches in this paper. Dasher is an dwell time free technology, which does not provide an eye-tracker. In the 9-direction Gaze Estimation technology the human gaze is divided into 9 sections and the user selects an section by looking in the right direction. The EyeSwipe system works like the FingerSwipe System on a smartphone: To type, the user ßwipes"his gaze across a keyboard. pEYEs is an eye typing system, that also tries to unify the input on a computer system. And iType is a system, that provides a secure way to input passwords on a smartphone using eye typing.

LITERATUR

[1] Anke Huckauf and Mario H. Urbina. 2008. Gazing with pEYEs: Towards a Universal Input for Various Applications. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 51–54. https://doi.org/10.1145/1344471.1344483

[2] Mohamed Khamis, Florian Alt, and Andreas Bulling. 2018. The Past, Present, and Future of Gaze-enabled Handheld Mobile Devices: Survey and Lessons Learned. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. ACM, New York, NY, USA, Article 38, 17 pages. https://doi.org/10.1145/3229434.3229452

[3] Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. 2016. EyeSwipe: Dwell-free Text Entry Using Gaze Paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1952–1956. https://doi.org/10.1145/2858036.2858335

[4] Z. Li, M. Li, P. Mohapatra, J. Han, and S. Chen. 2017. iType: Using eye gaze to enhance typing privacy. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*. 1–9. https://doi.org/10.1109/INFOCOM.2017.8057233

[5] Kseniia Palin, Anna Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers.. In *Proceedings of 21st International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI'19)*.

[6] Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. 2008. Now Dasher! Dash Away!: Longitudinal Study of Fast Text Entry by Eye Gaze. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 19–26. https://doi.org/10.1145/1344471.1344476

[7] Chi Zhang, Rui Yao, and Jinpeng Cai. 2018. Efficient Eye Typing with 9-direction Gaze Estimation. *Multimedia Tools Appl.* 77, 15 (Aug. 2018), 19679–19696. https://doi.org/10.1007/s11042-017-5426-y